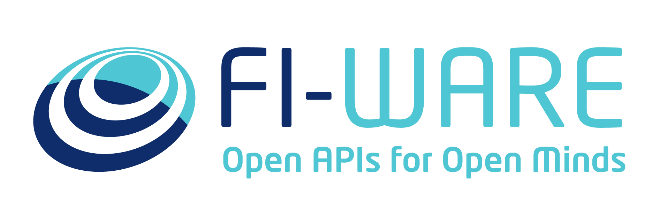
**MNCA IOT SmartCity Technical Architecture**

****

****

**DSI Métropole Nice Côte d’Azur**

**IOT & Smart City Platform**

|  |  |
| --- | --- |
| VERSION: draft | REVISION DATE: 02/11/2017 |

Approval of the Technical Architecture indicates an understanding of the purpose and content described in this deliverable. By signing this deliverable, each individual agrees with the content contained in this deliverable.

|  |  |  |  |
| --- | --- | --- | --- |
| **Approver Name** | **Title** | **Signature** | **Date** |
| Stéphane Roux | ICT Quality Assurance Engineer |  |  |
| Lionel Chaudanson | Responsable technique, IOT & Smart City |  |  |
| Serge Massiera | Directeur des Systèmes d'Information |  |  |
|  |  |  |  |
|  |  |  |  |

[1 Section 1 DOCUMENT SCOPE 6](#_Toc497654477)

[1.1 Context 6](#_Toc497654478)

[1.2 Glossary 8](#_Toc497654479)

[2 Section 2 OVERALL TECHNICAL ARCHITECTURE 9](#_Toc497654480)

[2.1 System Architecture Context Diagram 9](#_Toc497654481)

[2.2 System Architecture Model 11](#_Toc497654482)

[2.2.1 Overall Architectural Considerations 12](#_Toc497654483)

[2.3 System Architecture Component Definitions 15](#_Toc497654488)

[2.3.1 Security Component IDM GE 15](#_Toc497654489)

[2.3.2 Orion Context Broker Component 15](#_Toc497654491)

[2.3.3 MongoDB Cluster Component 15](#_Toc497654492)

[2.3.4 Citus/PostgreSQL Cluster Component 15](#_Toc497654493)

[2.3.5 Galera/MySQL Cluster Component 16](#_Toc497654494)

[2.3.6 Cosmos Big Data Analytics Component 16](#_Toc497654495)

[2.3.7 Knowage Analytics Component 16](#_Toc497654496)

[2.3.8 CKAN Open Data Component 17](#_Toc497654497)

[2.3.9 IDAS Component 17](#_Toc497654498)

[2.3.10 CEP Component 18](#_Toc497654499)

[2.3.11 Application Mashup Components 18](#_Toc497654500)

[2.3.12 Business Ecosystem Component 19](#_Toc497654501)

[2.3.13 Development Components 19](#_Toc497654502)

[2.3.14 Monitoring Components 20](#_Toc497654503)

[2.3.15 Docker Images Management Component 20](#_Toc497654504)

[2.3.16 Docker Swarm Management Component 20](#_Toc497654505)

[3 Section 3 SYSTEM ARCHITECTURE DESIGN 20](#_Toc497654506)

[3.1 Security Component IDM GE 21](#_Toc497654507)

[3.1.1 Component Functions 21](#_Toc497654508)

[3.1.2 Technical Considerations 21](#_Toc497654509)

[3.1.3 Selected Product(s) 21](#_Toc497654510)

[3.1.4 Selection Rationale 21](#_Toc497654511)

[3.1.5 Architecture Risks 21](#_Toc497654512)

[3.2 Orion Context Broker Component 21](#_Toc497654513)

[3.2.1 Component Functions 21](#_Toc497654514)

[3.2.2 Technical Considerations 21](#_Toc497654515)

[3.2.3 Selected Product(s) 21](#_Toc497654516)

[3.2.4 Selection Rationale 21](#_Toc497654517)

[3.2.5 Architecture Risks 21](#_Toc497654518)

[3.3 MongoDB Component 22](#_Toc497654519)

[3.3.1 Component Functions 22](#_Toc497654520)

[3.3.2 Technical Considerations 22](#_Toc497654521)

[3.3.3 Selected Product(s) 22](#_Toc497654522)

[3.3.4 Selection Rationale 22](#_Toc497654523)

[3.3.5 Architecture Risks 22](#_Toc497654524)

[3.4 Citus/PostgreSQL cluster Component 22](#_Toc497654525)

[3.4.1 Component Functions 22](#_Toc497654526)

[3.4.2 Technical Considerations 22](#_Toc497654527)

[3.4.3 Selected Product(s) 22](#_Toc497654528)

[3.4.4 Selection Rationale 22](#_Toc497654529)

[3.4.5 Architecture Risks 22](#_Toc497654530)

[3.5 Galera/MySQL cluster Component 22](#_Toc497654531)

[3.5.1 Component Functions 22](#_Toc497654532)

[3.5.2 Technical Considerations 22](#_Toc497654533)

[3.5.3 Selected Product(s) 22](#_Toc497654534)

[3.5.4 Selection Rationale 22](#_Toc497654535)

[3.5.5 Architecture Risks 22](#_Toc497654536)

[3.6 Cosmos Big Data Analytics Component 22](#_Toc497654537)

[3.6.1 Component Functions 23](#_Toc497654538)

[3.6.2 Technical Considerations 23](#_Toc497654539)

[3.6.3 Selected Product(s) 23](#_Toc497654540)

[3.6.4 Selection Rationale 23](#_Toc497654541)

[3.6.5 Architecture Risks 23](#_Toc497654542)

[3.7 Knowage Analytics Component 23](#_Toc497654543)

[3.7.1 Component Functions 23](#_Toc497654544)

[3.7.2 Technical Considerations 23](#_Toc497654545)

[3.7.3 Selected Product(s) 23](#_Toc497654546)

[3.7.4 Selection Rationale 23](#_Toc497654547)

[3.7.5 Architecture Risks 23](#_Toc497654548)

[3.8 CKAN Open Data Component 23](#_Toc497654549)

[3.8.1 Component Functions 23](#_Toc497654550)

[3.8.2 Technical Considerations 23](#_Toc497654551)

[3.8.3 Selected Product(s) 23](#_Toc497654552)

[3.8.4 Selection Rationale 23](#_Toc497654553)

[3.8.5 Architecture Risks 23](#_Toc497654554)

[3.9 IDAS Component 23](#_Toc497654555)

[3.9.1 Component Functions 23](#_Toc497654556)

[3.9.2 Technical Considerations 23](#_Toc497654557)

[3.9.3 Selected Product(s) 23](#_Toc497654558)

[3.9.4 Selection Rationale 23](#_Toc497654559)

[3.9.5 Architecture Risks 23](#_Toc497654560)

[3.10 CEP Component 24](#_Toc497654561)

[3.10.1 Component Functions 24](#_Toc497654562)

[3.10.2 Technical Considerations 24](#_Toc497654563)

[3.10.3 Selected Product(s) 24](#_Toc497654564)

[3.10.4 Selection Rationale 24](#_Toc497654565)

[3.10.5 Architecture Risks 24](#_Toc497654566)

[3.11 Application Mashup Component 24](#_Toc497654567)

[3.11.1 Component Functions 24](#_Toc497654568)

[3.11.2 Technical Considerations 24](#_Toc497654569)

[3.11.3 Selected Product(s) 24](#_Toc497654570)

[3.11.4 Selection Rationale 24](#_Toc497654571)

[3.11.5 Architecture Risks 24](#_Toc497654572)

[3.12 Business Ecosystem Component 24](#_Toc497654573)

[3.12.1 Component Functions 24](#_Toc497654574)

[3.12.2 Technical Considerations 25](#_Toc497654575)

[3.12.3 Selected Product(s) 25](#_Toc497654576)

[3.12.4 Selection Rationale 25](#_Toc497654577)

[3.12.5 Architecture Risks 25](#_Toc497654578)

[3.13 Development Components 25](#_Toc497654579)

[3.13.1 Component Functions 25](#_Toc497654580)

[3.13.2 Technical Considerations 25](#_Toc497654581)

[3.13.3 Selected Product(s) 25](#_Toc497654582)

[3.13.4 Selection Rationale 25](#_Toc497654583)

[3.13.5 Architecture Risks 25](#_Toc497654584)

[3.14 Monitoring Components 25](#_Toc497654585)

[3.14.1 Component Functions 26](#_Toc497654586)

[3.14.2 Technical Considerations 26](#_Toc497654587)

[3.14.3 Selected Product(s) 26](#_Toc497654588)

[3.14.4 Selection Rationale 26](#_Toc497654589)

[3.14.5 Architecture Risks 26](#_Toc497654590)

[3.15 Docker Images Management Component 26](#_Toc497654591)

[3.15.1 Component Functions 26](#_Toc497654592)

[3.15.2 Technical Considerations 26](#_Toc497654593)

[3.15.3 Selected Product(s) 26](#_Toc497654594)

[3.15.4 Selection Rationale 26](#_Toc497654595)

[3.15.5 Architecture Risks 26](#_Toc497654596)

[3.16 Docker Swarm manager Component 26](#_Toc497654597)

[3.16.1 Component Functions 26](#_Toc497654598)

[3.16.2 Technical Considerations 26](#_Toc497654599)

[3.16.3 Selected Product(s) 26](#_Toc497654600)

[3.16.4 Selection Rationale 26](#_Toc497654601)

[3.16.5 Architecture Risks 26](#_Toc497654602)

[4 Section 4 System Construction Environment 26](#_Toc497654603)

[4.1 Development Environment 27](#_Toc497654604)

[4.1.1 Developer Workstation Configuration 27](#_Toc497654605)

[4.1.2 Supporting Development Infrastructure Configuration 27](#_Toc497654606)

[4.2 QA Environment 27](#_Toc497654607)

[4.2.1 QA Workstation Configuration 27](#_Toc497654608)

[4.2.2 Supporting QA Infrastructure Configuration 27](#_Toc497654609)

[4.3 Acceptance Environment 27](#_Toc497654610)

[4.3.1 Acceptance Workstation Configuration 27](#_Toc497654611)

[4.3.2 Supporting Acceptance Infrastructure Configuration 27](#_Toc497654612)

# Section 1 DOCUMENT SCOPE

## Context

***Document Scope*** *describes the context and the goals of this document in a narrative.*

This document describes the Technical Architecture of the IOT Smart City Platform that:

* satisfies business requirements as documented in the roadmap (ref to provide),
* satisfies technical, operational and transitional requirements.

In the referenced roadmap document the needs has been described through the following global schema:

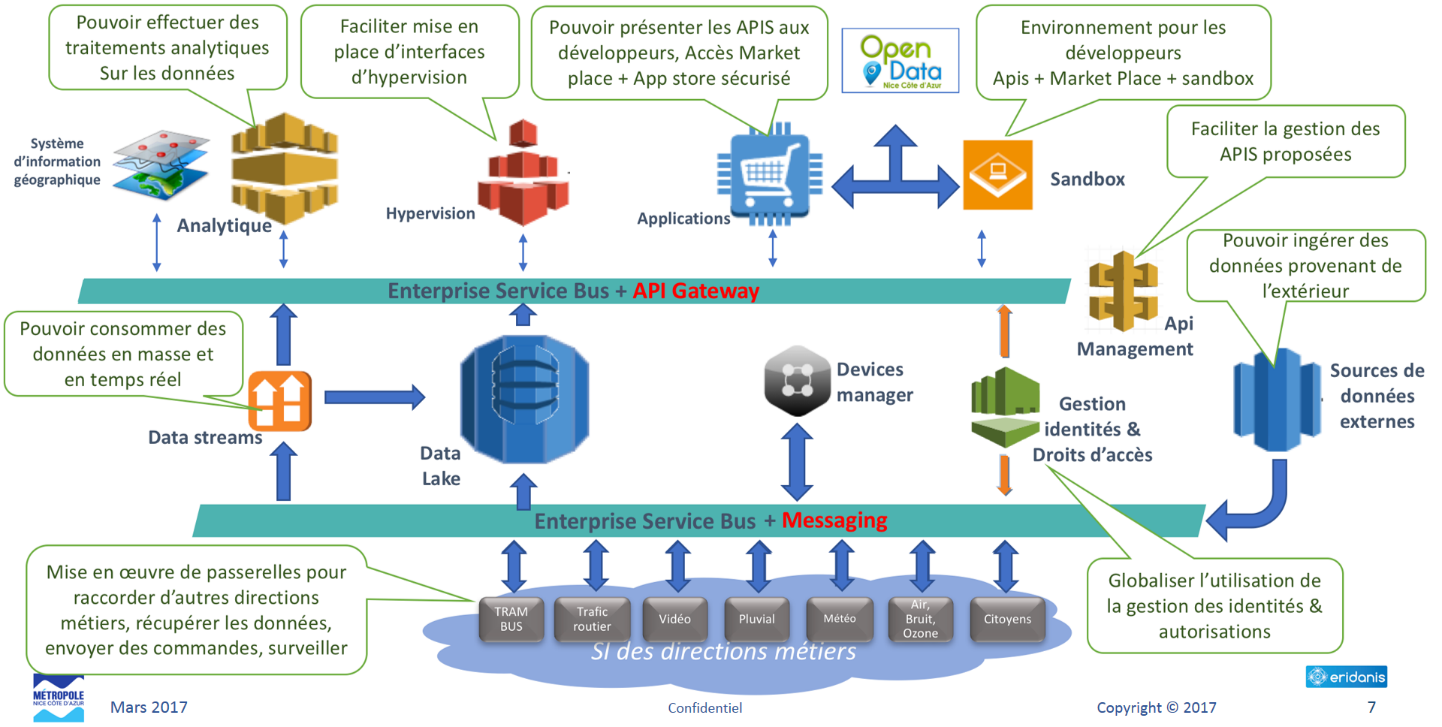


Figure 1: Target Logical Smart City Platform (TODO English version)

The goal of this Technical Architecture is to define the technologies, products, and techniques necessary to install and support the system, and to ensure that the system components are compatible and comply with the enterprise-wide standards and direction defined by Nice Côte d’Azur metropolis (hereafter referred as MNCA).

This document will also:

* Identify and explain the risks inherent in this Technical Architecture;
* Define baseline sizing, archiving and performance requirements;
* Identify the hardware and software specifications for the Development, Testing, QA and Production environments;
* Identify monitoring and support tools for maintenance in operational condition
* Define procedures for data migration among the environments (current datawarehouse).

The MNCA IOT Smart City Platform will be deployed upon the 3 data centers owned and managed by the MNCA DSI

* The main data center is Phoenix,
* The secondary is Bosio (currently underworks for revamping for 6 months),
* The third one (the smallest) is Biscarra, (the configuration has to be reconsidered but after Bosio .
* The hardware infrastructure is based on bare metal servers (Cisco and a few remaining HP)
* The servers are blades in enclosures, 3 enclosures per cabinet
* The machines are managed with VMWare ESX as virtualization layer, vCenter, vMotion.
* There is a redundant SAN (EMC), dispatched in main and secondary DC.
* It exists also a redundant NAS system for low I/O file storage.

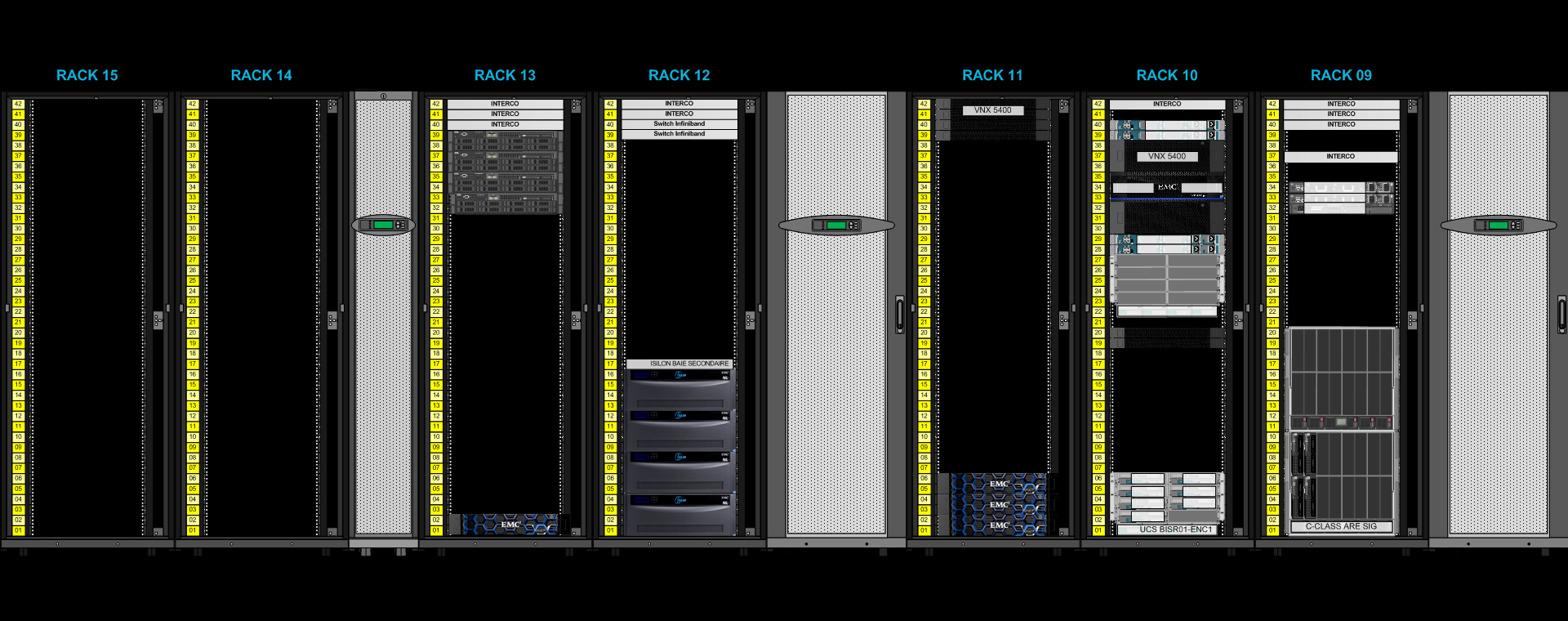
MNCA Network is managed with a cluster of firewall/routers, managing the separation in 4 different virtual networks

* Public DMZ for applications
* Private DMZ
* LAN
* Field LAN for IOT gateways, devices and sensors

There’s several Load balancing F5 Big IP appliances at strategic points to distribute load to data centres.

On next page is presented the overview of the Phoenix main Data Centre.







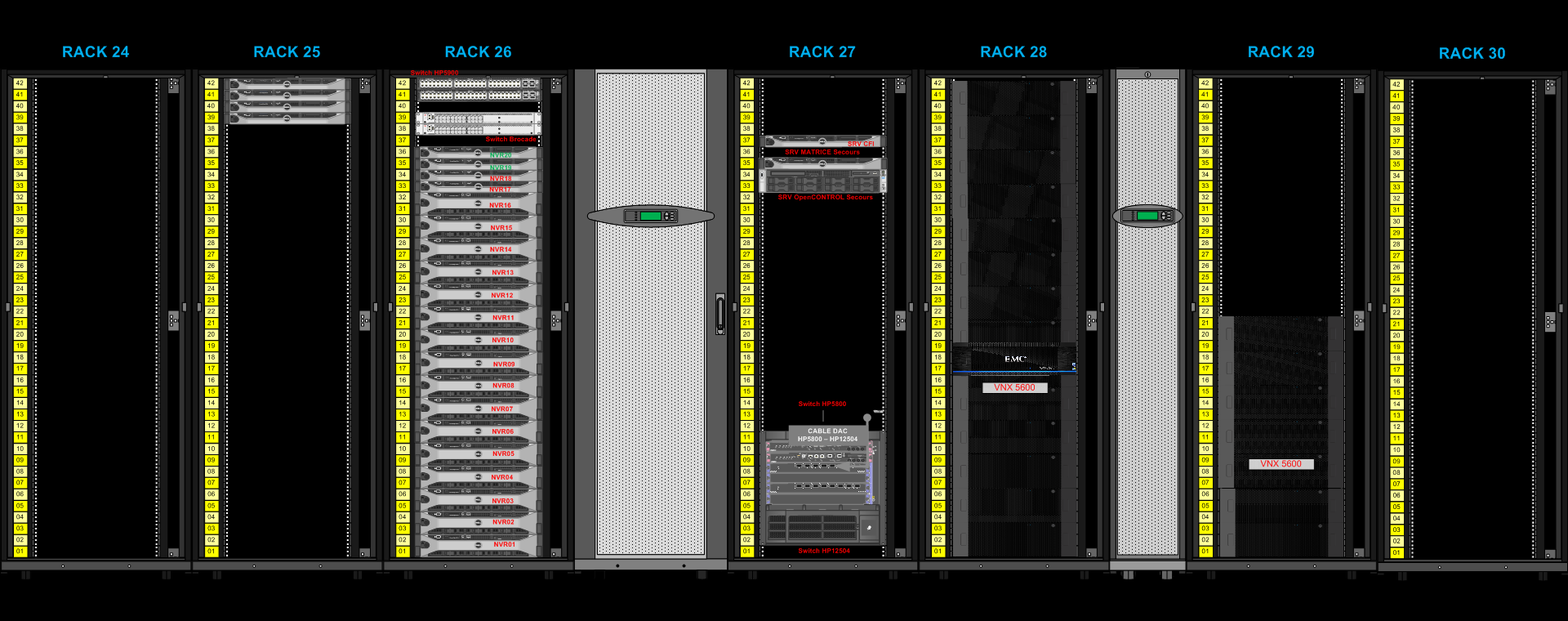


Figure 2: Poenix DC overview

## Glossary

**FIWARE**: is an independent open community whose members are committed to materialize the FIWARE mission, that is: “to build an open sustainable ecosystem around public, royalty-free and implementation-driven software platform standards that will ease the development of new Smart Applications in multiple sectors”.

**Fiware platform**: is a platform providing a simple but powerful set of APIs that ease the development of Smart Applications in multiple vertical sectors: <https://www.fiware.org/>

**IOT**: Internet Of Things

**OMA**: [Open Mobile Alliance](http://www.openmobilealliance.org/) : OMA is a non-profit organization that delivers open specifications for creating interoperable services that work across all geographical boundaries, on any bearer network.

**TM Forum**: is the global industry association that drives digital transformation of the communications industry through collaboration.

**GE**: Generic Enabler, Fiware used terminology to design an essential module of the platform

**API**: Application Programming Interface

**Context Broker**: According to Gartner:

“A context broker is a service that is designed to gather reachable context data of a variety of types, sources and velocity. It then applies conditioning, integration, rules and analytics to derive the reduced prepared context data, actionable at a point of business decision by a system or a human.”

Functionally speaking, context enables the data to be empowered by the interaction with other pieces of data. In a nutshell it can be consider as a data exchange bus, providing publish/subscribe of context information.

**NGSI**: New Generation Services Interface, [OMA referenced API specifications](http://www.openmobilealliance.org/release/NGSI/) defined ([NGSI V9 and V10](https://forge.fiware.org/plugins/mediawiki/wiki/fiware/index.php/NGSI-9/NGSI-10_information_model)) and used with amendments by Fiware (Context Broker) as the dorsal for the platform

**Docker**: is a virtualization software technology providing containers to run software applications in isolation:

**Swarm**: Docker Swarm provides native clustering functionality for Docker containers, which turns a group of Docker engines into a single, virtual Docker engine.

**VRF**: Virtual Routing and Forwarding network component.

**GDPR**: General Data Protection Regulation

**RIA**: Rich Internet Application

# Section 2 OVERALL TECHNICAL ARCHITECTURE

## 2.1 System Architecture Context Diagram

The **Smart city plarform Architecture Context Diagram** provides the “big picture” view of the global system’s architecture, and puts it in context with the rest of the Performing Organization’s systems portfolio, illustrating how the system’s hardware and software platforms fit into the existing environment.



Figure 3: MNCA Infrastructure Global View

The above schema shows globally the IT and network infrastructure in place for MNCA.

Since servers are dispatched through the 4 VLANs independently from the cabinets, the schema is not the true image of cabinets repartition in the different VLANs

The existing IT services for MNCA are deployed through this available infrastructure.

The Smart city platform will as well be deployed on the same data center infrastructure, but a new DMZ network

Nevertheless, it might be that some new hardware will be required (servers, router/firewall, Load balancer)

The envisaged interaction between those services and the smart City platform is for the moment limited to:

* use MNCA mail services (outgoing/incoming)
* connect existing data sources to the new system instead of the current used data warehouse
* potentially use global MNCA security system based on Microsoft Adam (LDAP server), if necessary

After a selection process it has been agreed that the MNCA smart city platform will be mainly based on [FIWARE](https://www.fiware.org/)

The Fiware community propose a set of modules called Generic Enablers, which provide the essential software bricks to set a complete smart city platform, like a kind of “Lego”.

The FIWARE architecture is by nature open, and based on open source modules available in a [dedicated catalog](https://catalogue.fiware.org/)

The most important to mention is that FIWARE has deliberately chosen to go towards a container based architecture, this remove the dependency with Openstack initially thought as the preferred cloud stack for deploying Fiware platforms.

Fiware started in 2014, which was just the beginning of the “Dockerization” of the server software world, the Fiware consortium couldn’t stay without going towards container based deployment, Docker is now an officially supported way to deploy Fiware components in a very clear and easy way.

MNCA have the opportunity to introduce new way to use its infrastructure using container based architecture, by choosing to use Docker to deploy its own Fiware plarform as a reference Smart City Platform.

The global architecture of the MNCA Fiware platform will rely on a Docker Swarm composed of nodes dispatched in the different VLANs.

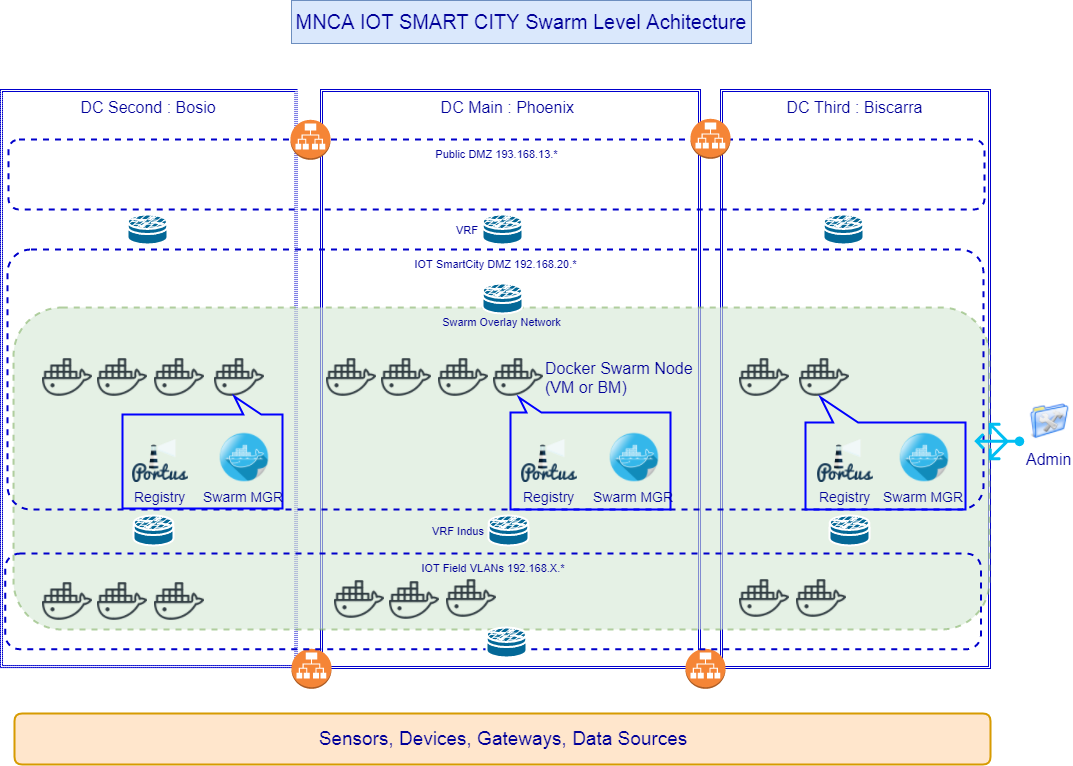


Figure 4: Docker swarm deployment for MNCA Smart city Fiware platform

A Docker swarm solution has been chosen in a first step because of the ease of use and configuration.

The learning curve of Docker container orchestration tool: Swarm, is less sloppy than other solution like Kubernetes (the open source declination of the Google internal orchestration tool: Borg), which sit at the top in term of features, openness, and adoption, among cloud orchestration tools today’s landscape.

Even it is the right choice to start with simple Docker Swarm, it will be easy to add later another orchestration tool like Kubernetes, since it will be proposed as the Swarm complement in Docker Enterprise distribution ([very recent news](https://thenewstack.io/docker-fully-embraces-kubernetes/)), the upgrade feasibility is guaranteed.

Then the MNCA FIWARE smart city platform will be deployed in a global docker swarm covering the 3 data centers.

The repartition of the traffic between the data centers will be managed by F5 BigIP load balancing appliance.

However, to minimize configuration task on load balancer (limit to load repartition between data centers of main categories of services), for the lower granularity of applications, and internal components of the platform, simple load balancing and fail-over mechanisms are managed automatically by the Docker Swarm (number of instances of a service).

## System Architecture Model

The **System Architecture Model** represents the various architecture components that comprise the system, and shows the important interrelationships.

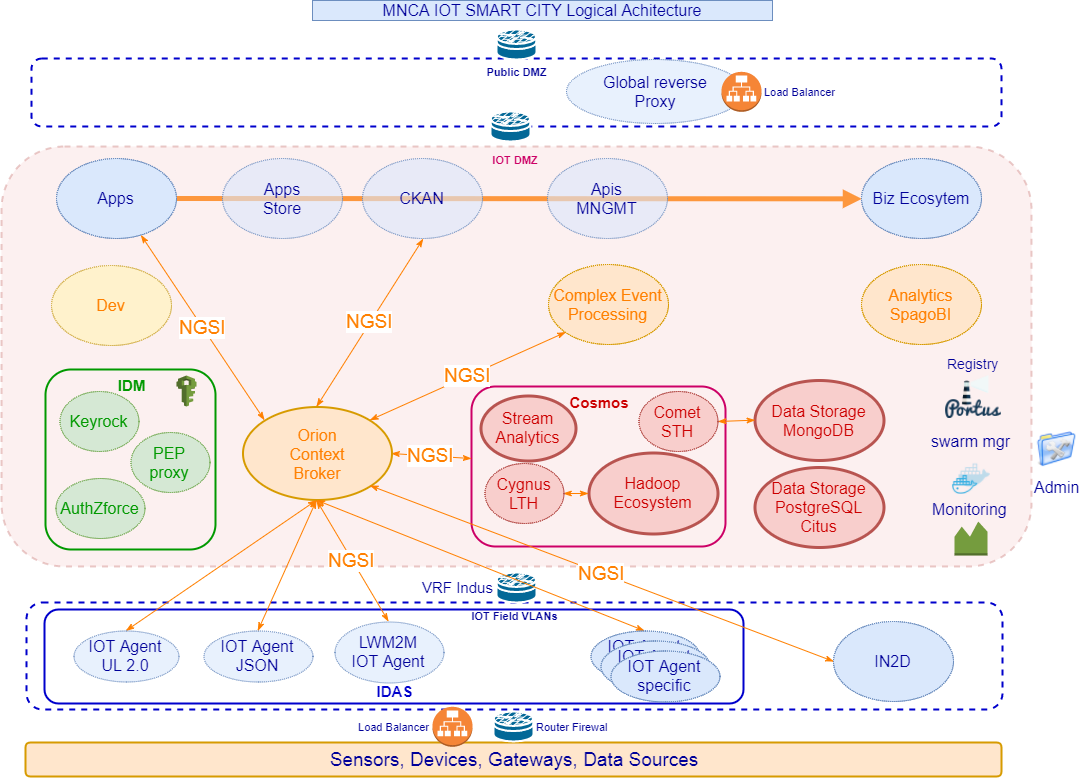


Figure 5: MNCA IOT Smart City Platform logical architecture

* Not all the relations between components have been represented in the above schema
* The central component that provides the main “super data bus” is the Orion Context Broker, this is a Fiware General Enabler (GE)
* As any interaction with this component is done using NGSI REST APIs, this is the main communication stream in the platform, that is represented on the above figure.
* As well the transversal arrow to Biz ecosystem GE, represent the link of each component to it for platform assets catalog management and monetization.
* Are not represented the relations to the identity manager (IDM GE) component, which are generally used to authorize access to the component through a proxy to identity manager.
* The aim with the IDM component, is to try to offer a centralized SSO management over the whole platform, allowing the central control of access authorization to platform components.
* The data storage component is not a Fiware component, this should be the SQL data tank of any Fiware module using SQL database, in case it’s impossible to change (a lot of Fiware modules rely on MySQL database) to PostgreSQL, we can use MariaDB.
* The general idea with the data storage components in the representation is to emphasis the fact that all the databases used by Fiware modules should be centralized in some clusters based on needed database server, which is true for SQL database and NoSQL database, so that relation links have not been represented on the above schema
* Inside the DEV component on the above figure are the Paas and the tools needed to develop applications that will use the Fiware ecosystem, this encompass development tools but also QA tools for testing and validation purpose, which can have specific needs in term of access to data. Most probably for development and QA, it will be required to deploy some Fiware components or even a complete platform for staging.
* A global reverse proxy and load balancer (F5 BigIP) will be the main entry point to the smart city platform.

### 2.2.1 Overall Architectural Considerations

*The* ***Overall Architectural Considerations*** *section defines how additional technical requirements have been addressed by the architecture. items in this section include:*

* *Deployment strategy*
* *Security Strategy*
* *Availability requirements*
* *Performance requirements*
* *Accessibility*
* *Database management*
* *Transaction volumes*
* *Concurrent users & devices*
* *Data import and export*
* *Disaster recovery*
* *GDPR*

#### Deployment Strategy

As stated previously, the MNCA Smart City platform will be deployed using Docker containers in a global swarm (group of Docker engines).

In this first step Docker engines will be deployed on Virtual machines dispatched on the 3 data centers (when they will be available).

The different modules will be grouped in services and stacks (Docker concepts).

Only Database servers, will be deployed in clusters, at disposal for use by other components of the platform.



#### Security Strategy

Security is achieved through a global identity and access rights management component the Fiware Identity Management GE, providing SSO and secure access to other components of the platform, according to requester authorizations.

#### Availability requirements

All the components used in the platform will be deployed in a way providing high availability.

The services and the associated databases should ideally be deployed on the 3 data centers,

The third data center remaining off-line most of time, should host replicas of the databases, updated in real-time.

The use of Docker provides an easy way to deploy highly available services, by deploying several containers on several nodes of a swarm, using docker compose files or docker stack files.

#### Performance requirements

Any of the components will be scalable by adding some instance of containers to scale-up the service.

Through the use of Docker it is very easy to achieve by changing services configuration on the fly.

#### Accessibility

Nothing specific is included in Fiware for accessibility, this item is to be considered at application UI level, most probably using accessibility features from operating systems or internet browsers that host the client application.

#### Database management

Different kind of databases are used among the Fiware components:

PostgreSQL, MySQL, MongoDB, and an Hadoop cluster.

To simplify the maintenance:

PostgreSQL should replace MySQL in each component where it’s possible, if it’s not possible MariaDB should replace MySQL without change.

To rationalize the architecture, each kind of database service will be deployed once in the platform, shared by the client components.

Each of the database service will be based on a cluster:

* This is native architecture for MongoDB
* PostgreSQL databases will be based on a Citus cluster
* MySQL databases will be based on a Galera cluster

May be thinking to master data management at the occasion of this project is an opportunity to consider

In particular to reach GDPR conformity more easily, by using appropriate tools to manage data and the access rights.

#### Transaction volumes

No specific requirement, best effort will be observed, but as the database clusters will be easily scalable, the transaction volume increase can be anticipated.

For that the monitoring system must have detailed history about transaction activity to be compared to processing/storage/network activity of the system, in order to be able to analyze the need of adding a node to the cluster, or just another container.

#### Concurrent user & devices

No specific requirement.

The scalability of the platform allows to virtually support any numbers of concurrent users and devices, providing the provisioning of enough computing nodes and bare metal servers to absorb the load.

#### Data import, export and models

Data existing in the current data warehouse will be imported into the new platform,

The main concern remains the current data model adaptation to existing [Fiware data models](https://www.fiware.org/data-models/) and eventually propose amendments when needed.

This is currently under study, in order to be able to propose a table of correspondence between the existing data source attributes and the attributes defined in the schema for the destination data object (an NGSI entity), that, if possible, match an existing Fiware data model.

In case new attributes need to be added to a Fiware proposed model, or new data model, the proposal should be sent to Fiware for validation, respecting the given guidelines on [How To Contribute](https://github.com/fiware/dataModels#how-to-contribute).

#### Disaster recovery

The 3 data centers should be involved in the smart city platform, then in case of serious problem in the main and secondary data center, the third DC, more protected from disaster like flood, can take over, and keep the platform online.

The databases used within the platform, must be replicated in real time on the third data center, to be able to use it in the shortest time possible.

#### GDPR

Observing General Data Protection Regulation will be mandatory on May 25th 2018, it’s a recent European directive defining new rules for personal data management.

Here are the major changes that are mentioned in this new legislation:

* **Consent:** Consent of personal data must be freely given, specific, informed and unambiguous. Consent is not freely given if a person is unable to freely refuse consent without detriment.
* **Accountability and privacy by default:** The GDPR has placed great emphasis on the accountability for **data controllers** to demonstrate data compliance. They will be required to maintain certain documentation, conduct impact assessment reports for riskier processing and employ data protection practices by default, such as data minimisation.
* **Notification of a data breach: Data controllers** must notify the Data Protection Authorities as quickly as possible, where applicable within 72 hours of the data breach discovery.
* **Sanctions:** This new legislation allows the Data Protection Authorities to impose higher fines – up to 4% of annual worldwide turnover. The maximum fines can be applied for discrepancies related to international data transfers or breach of processing principles, such as conditions for consent. Other violations can be fined up to 2% of annual worldwide turnover.
* **Role of data processors: Data processors** will now have direct obligations to implement technical and organisation measures to ensure data protection, this could include appointing a **Data Protection Officer** if needed.
* **One stop shop:** This legislation will be applicable in all EU states without the need of implementing national legislation. Having a single set of rules will benefit businesses as they will not need to comply with multiple authorities, streamlining the process and saving an estimate of €2.3 billion a year.
* **Removal of notification requirement: T**he requirement of notifying or seeking approval from a Data Protection Authority is going to be removed in many circumstances. This decision is made to save funds and time. Instead of notification the new directive requires **data controllers** to put in place appropriate practices for large scale processing in the form of new technology.
* **Right to be forgotten:** Persons will be able to require their data to be deleted when there is no legitimate reason for an organization to retain it. Following, the organisation must also take appropriate steps to inform any third party that might have any links or copies of the data and request them to delete it.
* **Expanded territorial reach:** Companies that are based outside of the EU, but targeting customers that are in the EU will be subject to the GDPR which is not the case now. (Not Applicable for the project).

Data Protection Act identifies Data controllers and Data Processors as separate jobs:

<https://ico.org.uk/media/for-organisations/documents/1546/data-controllers-and-data-processors-dp-guidance.pdf>

The new GDPR rules described above will have to be applied mostly in the applications involving the use of personal data (in particular data from citizens).

However, as the platform will be recipient of many different kind of data, it could be hard to embrace the complexity of the data sets possible entanglement, some data can be related to citizens, because coming from objects/sensors at home, even weared, or simply published through applications.

Then to have a Data Protection office is worth to consider (if not already acted), but this is beyond the scope of this document.

The multitenant nature of the Fiware platform help a lot to data protection & privacy by dividing into independent services with secure and controlled (authorized and authenticated) access to data-sets, provided by global security IDM component.

However, in the case of cross usage of multiple history datasets like in some data-science cases, data must be anonymized and minimized, before being delivered for analysis, this is to consider either at data acquisition step and data consuming step.

## System Architecture Component Definitions

### Security Component IDM GE

This component is the Fiware GE responsible for Fiware platform global user identity & access rights management

It is composed of 3 elements:

* the Identity Manager GE, in fiware catalog: [Keyrock](https://catalogue.fiware.org/enablers/identity-management-keyrock)
* the Access Control GE, in fiware catalog: [AuthZForce](https://catalogue.fiware.org/enablers/authorization-pdp-authzforce)
* GE module to add security to backend applications, in fiware catalog: [PEP Proxy](https://catalogue.fiware.org/enablers/pep-proxy-wilma)

|  |  |
| --- | --- |
| **Architecture Component** | **Component Elements** |
| Database Server | changed to Postgresql/Citus cluster instead of MySQL |
| Applications | Keyrock (API backend + front-end: Horizon Django based application)  AuthZforce  PEP Proxy |

Applications will be deployed through a service (in a Docker Stack file) providing high-availability and easy scalability.



### Orion Context Broker Component

This component is central to the Fiware platform: this is the data exchange bus of the platform.

The reference implementation of the Publish/Subscribe Context Broker GE.

Orion Context Broker allows to manage the entire lifecycle of context information including updates, queries, registrations and subscriptions.

It is an [NGSIv2](http://fiware.github.io/context.Orion/api/v2/stable/) server implementation to manage context information and its availability.

On Fiware catalog: [Orion Context Broker](https://catalogue.fiware.org/enablers/publishsubscribe-context-broker-orion-context-broker)

On Readthedocs: [Orion documentation](https://fiware-orion.readthedocs.io/en/master/)

|  |  |
| --- | --- |
| **Architecture Component** | **Component Elements** |
| Database Server | MongoDB |
| Application | Orion context broker |

### MongoDB Cluster Component

This component is the [MongoDB](https://www.mongodb.com/) NoSQL database cluster global to the platform.

Several Fiware components use MongoDB to persist data, among them: Orion, IDAS, Comet STH.

Clustering is native in MongoDB, there’s a master, and slaves,

The component is deployed as a cluster of MongoDB nodes, as a Docker service through a Stack file.

### Citus/PostgreSQL Cluster Component

This component is the cluster of [PostgreSQL](https://www.postgresql.org/) database servers of the Fiware platform.

The component is based on [Citus](https://www.citusdata.com/product/community), an open source clustering solution for PostgreSQL.

PostgreSQL is used by several components in the platform: such as CKAN, and those that will be changed from MySQL to PostgreSQL like Keyrock, Biz Eco System.

In Citus architecture, the manager is a specific PostgreSQL database server instance that needs a replication to insure manager fail-over and provides complete high availability.

Several replication solutions exist for PostgreSQL, the choice of a particular one is not yet decided

### Galera/MySQL Cluster Component

This component should be the cluster of MySQL database servers of the Fiware platform.

This component will be installed only if changing from mySQL to PostgreSQL for a module is too complex or impossible.

Any MySQL database that can’t be replaced with PostgreSQL, should be replaced by MariaDB,

To provide high availability and scalability the component will be based on a MySQL/MariaDB clustering solution: [Galera cluster](http://galeracluster.com/products/).

There is a MariaDB and Galera integration chapter on [MariaDB web site](https://mariadb.com/kb/en/library/galera-cluster/).

### Cosmos Big Data Analytics Component

This component is the Big Data and data history GE of the Fiware platform.

As the most complex component, it’s composed of many modules:

* Cygnus, as an interface between the context broker and the different data history storages
* Comet STH for Short Term History storage, uses MongoDB to persist the data sent through the context broker.
* An Hadoop ecosystem (HAAS) to store and process (batch or stream) long term history for analytics purposes.

Hadoop HDFS provides Distributed storage.

Distributed batch processing (with MapReduce).

Distributed stream processing through Sinfonier (Apache Storm based) or Spark.

Also, almost any software of the Hadoop ecosystem can be used in Cosmos: such as Hive, Pig, zookeeper, yarn, HBase…

On Fiware catalog: [Cosmos](https://catalogue.fiware.org/enablers/bigdata-analysis-cosmos)

On readthedocs: [Cosmos documentation](http://fiware-cosmos.readthedocs.io/en/latest/)

On Fiware Forge: [Architecture documentation](http://forge.fiware.org/plugins/mediawiki/wiki/fiware/index.php/FIWARE.ArchitectureDescription.Data.BigData)

|  |  |
| --- | --- |
| **Architecture Component** | **Component Elements** |
| Database Server | MongoDB cluster |
| Applications | Cygnus  Comet STH  Hadoop ecosystem: Sahara HAAS engine or HAAS light version engine  Tidoop (MapReduce)  Sinfonier (stream analytics Apache storm based) |

### Knowage Analytics Component

This component is for business intelligence, Fiware have originally chosen SpagoBI as the reference platform for this Analytics GE.

In the Fiware catalog is proposed the evolution of SpagoBI to a more integrated suite with all modules included Knowage

On Fiware catalog: [Knowage](https://catalogue.fiware.org/enablers/data-visualization-knowage)

On readthedocs: [Knowage in Fiware](http://knowage.readthedocs.io/en/latest/)

Knowage is a professional open source suite for modern business analytics over traditional sources and big data systems.

This GE provides the following features:

* BD (big data), to analyse data stored on big data clusters or NoSQL databases
* SI (smart intelligence), the usual business intelligence on structured data, but more oriented to self-service capabilities and agile prototyping
* ER (enterprise reporting), to produce and distribute static reports
* LI (location intelligence), to relate business data with spatial or geographical information
* PM (performance management), to manage KPIs and organize scorecards
* PA (predictive analysis), for more advanced analyses
* EI (embedded intelligence), to link Knowage with external solutions provided by the customer or third parties.

Knowage can connect to CKAN and show the catalog of data sets in it, allows to search in it, in order to be able to use it (only CSV & XLS data source for the moment, other formats will be added later).

Knowage can also define an NGSI dataset (query data from the Orion Context Broker

It will be configured to use the Keyrock IDM component for authentication.

|  |  |
| --- | --- |
| **Architecture Component** | **Component Elements** |
| Database Server | PostgreSQL/Citus for metadata storage |
| Applications | A set of java modules deployed under a Tomcat |

The composer part of Knowage, shouldn’t be exposed to public use, if some dashboards or reports visualization needs to be published, it need to be integrated in applications in an iframe component or by using the API to get the Knowage generated content.

### CKAN Open Data Component

This component is the chosen Fiware GE to manage Open Data.

CKAN GE at Fiware: <https://catalogue.fiware.org/enablers/ckan>

Fiware Docker installation: <https://github.com/okfn/docker-fiware-ckan>

Official installation guide : <http://docs.ckan.org/en/ckan-2.3.5/maintaining/installing/install-using-docker.html>

The CKAN GE provides the following features:

* Complete catalog system with easy to use web interface and a powerful API
* Strong integration with third-party CMS’s like **Drupal** (currently in use at MNCA) and WordPress
* Data visualization and analytics
* Workflow support lets departments or groups manage their own data publishing
* Fine-grained access control
* Integrated data storage and full data API
* Federated structure: easily set up new instances with common search

|  |  |
| --- | --- |
| **Architecture Component** | **Component Elements** |
| Database Server | Postgresql/Citus cluster |
| Applications | Api and WEB UI |

### IDAS Component

The IDAS component is an implementation of the Backend Device Management GE.

It provides software for IOT use cases, like:

* to manage communication between sensors&actuators/devices/gateways and the data context broker using NGSI api.
* to manage provisioning of devices through a REST API, it is composed of several components organized by south bound protocols (UL2, JSON, LWM2M, Sigfox)

On Fiware Catalog: [IDAS](https://catalogue.fiware.org/enablers/backend-device-management-idas)

Documentation on the Fiware IOT stack : [by Telefonica](https://fiware-iot-stack.readthedocs.io/en/latest/)

|  |  |
| --- | --- |
| **Architecture Component** | **Component Elements** |
| Database Server | MongoDB |
| Applications | [UL2.0 IOT Agent](https://github.com/telefonicaid/iotagent-ul)  [JSON IOT Agent](https://github.com/telefonicaid/iotagent-json)  [OMA LWM2M IOT Agent](https://github.com/telefonicaid/lightweightm2m-iotagent)  [SIGFOX IOT Agent](https://github.com/telefonicaid/sigfox-iotagent)  [IOT Agent library to develop other agents](https://github.com/telefonicaid/iotagent-node-lib)  IOT Agent Manager (under upgrade for R5 release)  IoT Edge Manager (forthcoming for R5)  [IOT Orchestrator with portal UI](https://github.com/telefonicaid/orchestrator) (from Telefonica)  MQTT broker like [Mosquitto](http://mosquitto.org/) is needed for MQTT connectivity of IOT agents. |

As the IOT agent components provide only a REST API to interact with, it lacks a web user interface for device management in the current Fiware catalog.

The [IOT Agent Manager](https://github.com/telefonicaid/iotagent-manager) is under porting in Nodejs for release 5, it should provide the device management layer.

Most of the IOT modules in Fiware catalog come from [Telefonica, in their github](https://github.com/telefonicaid), they provide a component called [Orchestrator](https://github.com/telefonicaid/orchestrator), that allows to interact with the different modules: IOT agents, perseo CEP, Orion, IDM, they also provide a web portal ready to use.

### CEP Component

The CEP component is the Complex Event Processing GE from the Fiware platform.

In Fiware catalog is proposed an implementation from IBM of the CEP module: [PROTON](https://catalogue.fiware.org/enablers/complex-event-processing-cep-proactive-technology-online)

It is a scalable integrated platform to support the development, deployment, and maintenance of event-driven applications.

Telefonica propose also an implementation of the CEP GE: [Perseo](http://fiware-iot-stack.readthedocs.io/en/latest/cep/), as the CEP GE of their IOT stack

As well as Orange which propose [Cepheus](https://catalogue.fiware.org/enablers/iot-data-edge-consolidation-ge-cepheus)

It’s composed of a light NGSI broker and a CEP which features a dedicated REST management API to configure real time analysis on NGSI events.

The CEP from Telefonica **Perseo** has been chosen for the MNCA platform, because it seems simpler to use at a first glance (in addition Telefonica IOT portal provides provisioning UI to Perseo).

|  |  |
| --- | --- |
| **Architecture Component** | **Component Elements** |
| Database Server | MongoDB |
| Applications | Perseo core engine (rules processing)  Perseo Front-end (Rules definition & storage) |

### Application Mashup Components

Wirecloud is the Fiware GE to build Rich Internet Applications (RIA), using semantic technologies to offer a next-generation end-user centered web application mashup platform.

On Fiware Catalog: [Wirecloud](https://catalogue.fiware.org/enablers/application-mashup-wirecloud)

On Readthedocs : [WirecloudDocumentation](https://wirecloud.readthedocs.io/en/stable/)

|  |  |
| --- | --- |
| **Architecture Component** | **Component Elements** |
| Database Server | Postgresql/Citus |
| Applications | Web application  back-end server |

It is a web application mashup platform aimed at allowing end users without programming skills to easily create web applications and dashboards/cockpits.

It integrates completely to Fiware architecture, by connecting directly to Context Broker and history storage tools.

Mashups and Dashboards made with Wirecloud, can be monetized through the business ecosystem component

Alternatively a small javascript library like [Freeboard](https://github.com/Freeboard/freeboard) can be used to visualize data coming from NGSI context broker: [news on Fiware](https://www.fiware.org/2015/07/13/fiware-orion-data-source-now-available-for-freeboard/). (an [Orion FIWARE datasource](https://github.com/telefonicaid/fiware-connectors/tree/master/FreeBoard-Orion-Plugin) has been developed by Telefonica)

Freeboard can be used also from web where you can publish dashboards: [Freeboard.io](https://freeboard.io/)

It can also be deployed on a web server, on the platform, but there’s no security or publishing management provided, it has to be added, by passing through a proxy.

As it’s pure javascript library and some json definition and configuration, it can be integrated in a web application that is proposed in the market place.

### Business Ecosystem Component

The Business API Ecosystem exposes its complete functionality through TM Forum standard APIs; concretely, it includes the following key features:

* Support for the management of catalogs, products, and offering
* Support for rich pricing models, including recurring payments, pay-per-use, etc.
* Support for accounting callbacks
* Support for billing and charging
* Integrated support for PayPal, including customer charges and seller payments
* Support for revenue sharing, including models with multiple stakeholders involved

Link to Fiware catalog: [Biz Api Ecosystem](https://catalogue.fiware.org/enablers/business-api-ecosystem-biz-ecosystem-ri)

Link to the documentation: <http://business-api-ecosystem.readthedocs.io/en/latest/>

This component is the Market Place of the Smart City platform, it allows to leverage:

Authorization access to services, and their monetization, even with several stakeholders involved that share revenues,

services here includes the applications available in the catalog, the data sets in CKAN, the available APIs.

|  |  |
| --- | --- |
| **Architecture Component** | **Component Elements** |
| Database Server | Postgresql/Citus (change from MySQL) |
| Applications | Web application  back-end servers:   * TM Forum APIs reference implementation (catalog, ordering, inventory, usage, billing, party Apis) * Charging Backend, * Revenue Sharing Service (RSS) * Logic Proxy |

### Development Components

Fiware was proposing to use [FusionForge](https://fusionforge.org/) (the software suite originated from Forge that was running SourceForge.net), Eclipse as an IDE as the set of tools for development.

Now with the advent of Docker, Eclipse have evolved towards the container paradigm and propose [Eclipse Che](https://www.eclipse.org/che/) as an IDE & Developer Workspace server, in addition it provides a RESTful API to interact with.

If for many applications like dashboards, using Wirecloud component is a good solution, for more complex or even mobile application it is not adapted, so that Eclipse Che IDE is an appropriate solution to develop Docker ready applications.

Eclipse Che is the next generation of IDE, provides integrated connectors to git and SVN, so that it can be easily connected to the Gitlab of MNCA.

An Eclipse Che server can be set in a multi-users environment, in standard installation it uses Keycloack as IDM provider, and a PostgreSQL database for metadata.

As Keycloak can replace Keystone as an authentication service (OAuth provider), in OpenStack deployment, and Eclipse Che can also use OAuth from GitLab or WSO2, if needed Keyrock can be also used as the OAuth provider giving SSO to developers using the Eclipse Che server.

An Eclipse Che Server using the IDM component, is a good candidate for the Paas IDE for IOE of the MNCA smart city platform.

### Monitoring Components

[Portainer](https://portainer.io/) will be used for real time management & monitoring of services and the composing containers of the swarm.

Portainer on [Github](https://github.com/portainer/portainer/blob/develop/README.md).

Portainer will be replicated on several nodes with several instances, ideally on each swarm manager (mandatory to have vision of the swarm).

As Portainer don’t use a Database, but store it’s configuration in files, mainly for user and dependencies management, so that those configuration files should be stored in a secured NFS storage volume, shared by all instances.

Centreon is already in use at MNCA, containers should be instrumented with needed check modules to monitor services and nodes in the platform from the Centreon, an additional Centreon service could be added in the platform if needed.

One thing to think about is the expansion of the platform, in terms of numbers of services deployed, this will lead to complexify the task of monitoring configuration if it remains manual, so that it’s worth to think about automatic services discovery to simplify and automate monitoring for OPS

One solution should be to use Consul for services discovery.

### Docker Images Management Component

As MNCA needs to manage its own Docker images, [Portus](http://port.us.org/) has been chosen as the component for that task.

It’s an [open source software originated from Suse](https://github.com/SUSE/Portus), written in Ruby, providing its own security access layer

### Docker Swarm Management Component

Swarm manager is the orchestrator of the services based on the components of the platform.

It’s part of Docker software suite.

To insure high availability several docker nodes should be swarm managers, at the minimum one in each of the 3 DCs, there’s no real need of scalability, since it’s an internal orchestration management interface.

### Poi Data Provider Component

Fiware GE to provide Point Of Interest (POI) management.

On Fiware Catalog: [POI Data Provider](https://catalogue.fiware.org/enablers/poi-data-provider)

POI (Points of interest) Generic Enabler is a web server kit that supports

* storing information related to locations
* serving queries by location and other criteria
* can be configured to meet your data needs

|  |  |
| --- | --- |
| **Architecture Component** | **Component Elements** |
| Database Server | Postgresql/Citus + PostGIS  MongoDB |
| Applications | API back-end server  PHP web application |

### GIS Data Provider Component

Fiware GE to provide spatial data storage.

This GE is able to host geographical data and serve it in 3D form (where applicaple) to both mobile and web clients. The GE implementation is based on open source Geoserver project (GPL licenced) and W3DS (Web 3D Service) extension.

On Fiware catalog: [GIS data Provider](https://catalogue.fiware.org/enablers/gis-data-provider-geoserver3d)

On readthedocs: [GIS Data Provider](http://gisdataprovider.readthedocs.io/en/latest/index.html)

|  |  |
| --- | --- |
| **Architecture Component** | **Component Elements** |
| Database Server | Postgresql/Citus + PostGIS |
| Applications | GeoServer |

# Section 3 SYSTEM ARCHITECTURE DESIGN

*The* ***System Architecture Design*** *section provides detailed descriptions of each product implementing architecture components, and explains the rationale for product selection.*

*For each* ***System Architecture Component*** *(identified in Section 2.3 above), this section describes: specific* ***Component Functions****, requirements and other* ***Technical Considerations*** *that were used in the decision-making process, as well as any specific* ***Products*** *selected to implement this component. The* ***Selection Rationale*** *identifies any other products that may have been considered, and provides rationale for the decision.* ***Architecture Risks*** *identifies any potential risks associated with the architecture element.*

## 3.1 Security Component IDM GE

### Component Functions

Manage globally the security for the Fiware platform, in terms of identity and access authorization to services and data.

The Keyrock module provides identity management Keystone back-end, the integrated web UI is provided through Horizon front-end, which allows user accounts management.

The AuthZforce component provides roles and access rights management

The PEP Proxy components provides Keyrock security to others back’end of the platform such as Orion context broker

### Technical Considerations

This is the components proposed in Fiware catalog,

It’s based on **Keystone**, the IDM component of OpenStack cloud software suite.

it use the standard OAuth2 protocol

it can connect with an LDAP if needed

### Selected Product(s)

OpenStack **Keystone** provides authentication, authorization and service discovery mechanisms via HTTP primarily for use by projects in the OpenStack family. It is most commonly deployed as an HTTP interface to existing identity systems, such as LDAP.

### Selection Rationale

Based on OpenStack IDM component Keystone

Oauth2 support, openID…

### Architecture Risks

As a globally used component for Identification and access rights management, it is a highly critical component.

The database must be maintained in a clustered environment that insure high availability as well as scalability, this is provided through the Citus cluster of PostgreSQL databases or the Galera cluster of MariaDB.

The involved application modules (Keyrock,

## Orion Context Broker Component

### Component Functions

Manage context information exchange for the whole platform using NGSI/V2 Api, this is a core function of the Fiware Platform, as it provides the link between the data coming from things in the field, where data models are many, and their usage in the

### Technical Considerations

### Selected Product(s)

There’s another implementation of an NGSI context broker originated from Orange:

But tailored for use at IOT gateway level, for example to manage a network of sensors relative to a specific activity among numerous smart city use cases

### Selection Rationale

Orion is the reference component of NGSI Context Broker in Fiware

### Architecture Risks

This component is central to the platform, because it’s the dorsal of the architecture

It can become a bottleneck in case of high traffic, so a particular attention is needed on response time, through specific monitoring

## MongoDB Component

### Component Functions

MongoDB is a NOSQL document database, natively scalable by multiplying the nodes running the software and replicating the data upon the nodes

### Technical Considerations

### Selected Product(s)

### Selection Rationale

Choice as database engine for several components proposed in Fiware.

A very popular scalable open source document oriented NoSQL database

### Architecture Risks

When number of documents became large, and structure complexify, MongoDB engine can show some limits, notably read performance degrades with large number of complex documents.

For this reason, the usage of MongoDB in data history management, is limited to short term history in Fiware architecture.

For long term history storage Fiware advice an Hadoop as a service platform.

Anyway, by using a solution based on PostgreSQL, like Stampede for Mongo slaves, that could be mapped to a final PostgreSQL/Citus cluster, high performance can be achieved on data reading and writing, even with long term history.

In addition it can be simpler o manage than an Hadoop environment, this is a solution to consider.

## Citus/PostgreSQL cluster Component

### Component Functions

Citus provide a way to make a cluster of PostgreSQL database servers, it will serve database(s) to any component of the platform that need it.

### Technical Considerations

Probably the open source database that is the closest to Oracle in term of performance in traditional transaction based SQL applications, this at no cost.

But PostreSQL have also introduced NoSQL capabilities, that allows them to challenge tenors like MongoDB in the domain.

It is easy to add spatial capabilities to Postgresql, wit PostGIS, well known and very popular in the world of GIS.

### Selected Product(s)

### Selection Rationale

Probably the open source database tah

### Architecture Risks

## Galera/MySQL cluster Component

### Component Functions

### Technical Considerations

### Selected Product(s)

### Selection Rationale

### Architecture Risks

## Cosmos Big Data Analytics Component

### Component Functions

### Technical Considerations

### Selected Product(s)

### Selection Rationale

### Architecture Risks

## Knowage Analytics Component

### Component Functions

### Technical Considerations

### Selected Product(s)

### Selection Rationale

Knwoage takes part in the FIWARE community, being the reference implementation of the Data Visualization GE.

### Architecture Risks

## CKAN Open Data Component

### Component Functions

### Technical Considerations

### Selected Product(s)

### Selection Rationale

### Architecture Risks

## IDAS Component

### Component Functions

### Technical Considerations

### Selected Product(s)

### Selection Rationale

### Architecture Risks

## CEP Component

### Component Functions

### Technical Considerations

### Selected Product(s)

### Selection Rationale

### Architecture Risks

## Application Mashup Component

### Component Functions

### Technical Considerations

### Selected Product(s)

### Selection Rationale

### Architecture Risks

## Business Ecosystem Component

### Component Functions

The Business Ecosystem component is composed of several modules that cover the following features:

**TM Forum Apis:**

* Catalog management,
* Ordering management,
* Inventory management,
* Usage management,
* billing,
* customer,
* party APIs.

**Business Ecosystem Charging Backend:**

* Process the different pricing models, the accounting information, and the revenue sharing reports.
* Calculate amounts to be charged, charge customers, and pay sellers, using this above information.

**Business Ecosystem Revenue Sharing Service:**

Distributes revenues originated by the usage of a given service among the involved stakeholders.

It distributes part of the revenue generated by a service between the Business API Ecosystem instance provider and the Service Provider(s) responsible for the service, a service here refers to both final applications and backend application services.

**Business Ecosystem Logic Proxy:**

Is the endpoint for accessing the Business API Ecosystem:

* it orchestrates the APIs validating user requests, including authentication, authorization, and the content of the request from a business logic point of view.
* it serves a web portal that can be used to interact with the system.

This is the component that allow to monetize applications, APIs and also data, that are exposed through the smart city platform

### Technical Considerations

TM Forum APIs and Revenue Sharing Service requirements

* Java 8
* Glassfish 4.1
* MySQL 5.5 => to replace with PostgreSQL/Citus or use Galera Cluster and MariaDB if hard to change from MySQL to PostgreSQL

Charging Backend requirements

* Python 2.7
* MongoDB
* wkhtmltopdf

Logic Proxy requirements

* NodeJS 4.5.0+ (Including NPM)

### Selected Product(s)

The other components

### Selection Rationale

This is the reference component in Fiware Catalog,

The use of TM Forum APIs is a warranty of being able to conduct easily business relations with different skateh

### Architecture Risks

Is to be consider as a critical component, because it mediates the usage of all applications and their APIs.

## Development Components

### Component Functions

### Technical Considerations

### Selected Product(s)

### Selection Rationale

### Architecture Risks

## Monitoring Components

### Component Functions

### Technical Considerations

### Selected Product(s)

### Selection Rationale

### Architecture Risks

## Docker Images Management Component

### Component Functions

### Technical Considerations

### Selected Product(s)

### Selection Rationale

### Architecture Risks

## Docker Swarm Manager Components

### Component Functions

### Technical Considerations

### Selected Product(s)

### Selection Rationale

### Architecture Risks

# Section 4 System Construction Environment

*The* ***System Construction Environment*** *section details the various environments necessary to enable system construction and testing.*

## Development Environment

### Developer Workstation Configuration

### Supporting Development Infrastructure Configuration

## QA Environment

### QA Workstation Configuration

### Supporting QA Infrastructure Configuration

## Acceptance Environment

*For each environment necessary for system construction (****Development, QA*** *and* ***Acceptance****), provide detailed specifications for the* ***Workstation*** *and* ***Supporting Infrastructure*** *that will be used (including hardware, network and operating system requirements, all necessary installed packages and tools, and needed directory structures**that will be utilized to store all construction components).*

### Acceptance Workstation Configuration

### Supporting Acceptance Infrastructure Configuration